PALM Intranet

Application Number [ ] Submit

IDS Flag Clearance for Application 09544253

IDS Information

Content	Mailroom Date	Entry Number	IDS Review	Last Modified	Reviewer
M844	2006-04-25	95	Y ☑	2006-08-25 07:55:33.0	ALy
M844	2006-01-20	88	Y ☑	2006-03-07 08:47:56.0	ALy
M844	2005-06-13	82	Y ☑	2005-07-01 14:41:11.0	wmeredith
M844	2005-03-31	80	Y ☑	2005-04-26 10:30:12.0	wmeredith
M844	2005-01-24	78	Y ☑	2005-04-26 10:29:41.0	wmeredith
M844	2004-12-20	72	Y ☑	2005-02-11 14:44:49.0	wmeredith
M844	2004-09-21	68	Y ☑	2004-12-09 14:39:36.0	wmeredith
M844	2004-05-17	66	Y ☑	2004-12-09 14:39:05.0	wmeredith
M844	2004-07-01	50	Y ☑	2004-07-20 11:17:44.0	wmeredith
M844	2004-03-08	39	Y ☑	2004-03-24 15:26:07.0	wmeredith
M844	2004-03-08	38	Y ☑	2004-03-24 15:12:06.0	wmeredith
M844	2004-01-02	37	Y ☑	2004-01-13 08:51:14.0	cnorfleet
M844	2003-07-25	36	Y ☑	2004-01-02 14:35:41.0	cnorfleet
M844	2003-11-14	33	Y ☑	2003-12-09 16:10:01.0	wmeredith
M844	2003-06-27	28	Y ☑	2003-07-15 09:04:14.0	bharrison1
M844	2003-06-18	27	Y ☑	2003-06-25 17:01:41.0	wmeredith
M844	2003-02-27	16	Y ☑	2003-03-11 16:09:17.0	wmeredith
Update					

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+(classpath or "class path") +classes +search +cache

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **classpath** or **class path** **classes** **search** **cache**

Found **13** of **184,245**

Sort results by	relevance
Display results	expanded form

Save results to a Binder
Search Tips
□ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 13 of 13

Relevance scale

1 Full papers: Runtime aspect weaving through metaprogramming
Jason Baker, Wilson Hsieh
April 2002 **Proceedings of the 1st international conference on Aspect-oriented software development**
Publisher: ACM Press
Full text available: pdf(883.36 KB) Additional Information: full citation, abstract, references, citings, index terms

We describe an extension to the Java language, Handi-Wrap, that supports weaving aspects into code at runtime. Aspects in Handi-Wrap take the form of method wrappers, which allow aspect code to be inserted around method bodies like advice in AspectJ. Handi-Wrap offers several advantages over static aspect languages such as AspectJ. First, aspects can be woven into binary libraries. Second, a wrapper in Handi-Wrap is a first-class Java value, which allows users to exploit Java mechanisms to defin ...

2 OOPSLA onward!: Finding bugs is easy
David Hovemeyer, William Pugh
December 2004 **ACM SIGPLAN Notices**, Volume 39 Issue 12
Publisher: ACM Press
Full text available: pdf(506.92 KB) Additional Information: full citation, abstract, references

Many techniques have been developed over the years to automatically find bugs in software. Often, these techniques rely on formal methods and sophisticated program analysis. While these techniques are valuable, they can be difficult to apply, and they aren't always effective in finding real bugs.*Bug patterns* are code idioms that are often errors. We have implemented automatic detectors for a variety of bug patterns found in Java programs. In this paper, we describe how we have used bug pa ...

3 ObjectGlobe: Ubiquitous query processing on the Internet
R. Braumandl, M. Keidl, A. Kemper, D. Kossmann, A. Kreutz, S. Seltzsam, K. Stocker
August 2001 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 10 Issue 1
Publisher: Springer-Verlag New York, Inc.
Full text available: pdf(251.44 KB) Additional Information: full citation, abstract, citings, index terms

We present the design of ObjectGlobe, a distributed and open query processor for Internet data sources. Today, data is published on the Internet via Web servers which have, if at all, very localized query processing capabilities. The goal of the ObjectGlobe project is to establish an open marketplace in which *data* and *query processing capabilities* can be distributed and used by any kind of Internet application. Furthermore, ObjectGlobe integrates *cycle providers* (i.e., machi ...

Keywords: Cycle-, function- and data provider, Distributed query processing, Open systems, Privacy, Quality of service, Query optimization, Security

4 Exploiting on-line data sources as the basis of programming projects

Peter DePasquale

March 2006 **ACM SIGCSE Bulletin , Proceedings of the 37th SIGCSE technical symposium on Computer science education SIGCSE '06**, Volume 38 Issue 1

Publisher: ACM Press

Full text available: pdf(111.45 KB) Additional Information: full citation, abstract, references, index terms

The Internet is an exciting place to find real, interesting, and interactive sources of data for use in the classroom. This data (often real world), can provide the basis of interesting projects for CS1 and CS2 courses. Additionally, the source or exchange protocol can be used as a gentle introduction for novice computer science majors to the myriad of data formats and emerging technologies available today. Presented here are a number of interesting data sources used during the 2004-2005 academi ...

Keywords: CS1/CS2, on-line data sources, programming projects

5 Implementing jalapeño in Java

Bowen Alpern, C. R. Attanasio, Anthony Cocchi, Derek Lieber, Stephen Smith, Ton Ngo, John J. Barton, Susan Flynn Hummel, Janice C. Sheperd, Mark Mergen

October 1999 **ACM SIGPLAN Notices , Proceedings of the 14th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '99**, Volume 34 Issue 10

Publisher: ACM Press

Full text available: pdf(1.57 MB) Additional Information: full citation, abstract, references, citings, index terms

Jalapeño is a virtual machine for Java™ servers written in Java.A running Java program involves four layers of functionality: the user code, the virtual-machine, the operating system, and the hardware. By drawing the Java / non-Java boundary below the virtual machine rather than above it, Jalapeño reduces the boundary-crossing overhead and opens up more opportunities for optimization.To get Jalapeño started, a boot image of a ...

6 XML query and programming languages: XJ: facilitating XML processing in Java

Matthew Harren, Mukund Raghavachari, Oded Shmueli, Michael G. Burke, Rajesh Bordawekar, Igor Pechtchanski, Vivek Sarkar

May 2005 **Proceedings of the 14th international conference on World Wide Web**

Publisher: ACM Press

Full text available: pdf(293.50 KB) Additional Information: full citation, abstract, references, index terms

The increased importance of XML as a data representation format has led to several proposals for facilitating the development of applications that operate on XML data. These proposals range from runtime API-based interfaces to XML-based programming languages. The subject of this paper is XJ, a research language that proposes novel mechanisms for the integration of XML as a first-class construct into Java™. The design goals of XJ distinguish it from past work on integrating XML support into ...

Keywords: Java, XML, language design

7 Technical papers: dynamic program analysis: Tracking down software bugs using automatic anomaly detection
Sudheendra Hangal, Monica S. Lam
May 2002 **Proceedings of the 24th International Conference on Software Engineering**
Publisher: ACM Press
Full text available: pdf(1.30 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper introduces DIDUCE, a practical and effective tool that aids programmers in detecting complex program errors and identifying their root causes. By instrumenting a program and observing its behavior as it runs, DIDUCE dynamically formulates hypotheses of invariants obeyed by the program. DIDUCE hypothesizes the strictest invariants at the beginning, and gradually relaxes the hypothesis as violations are detected to allow for new behavior. The violations reported help users to catch soft ...

8 Programming languages for mobile code
Tommy Thorn
September 1997 **ACM Computing Surveys (CSUR)**, Volume 29 Issue 3
Publisher: ACM Press
Full text available: pdf(393.65 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Sun's announcement of the programming language Java more that anything popularized the notion of mobile code, that is, programs traveling on a heterogeneous network and automatically executing upon arrival at the destination. We describe several classes of mobile code and extract their common characteristics, where security proves to be one of the major concerns. With these characteristics as reference points, we examine six representative languages proposed for mobile code. The conclusion ...

Keywords: Java, Limbo, Objective Caml, Obliq, Safe-Tcl, distribution, formal methods, mobile code, network programming, object orientation, portability, safety, security, telescript

9 Distrbuted VEEs: PDS: a virtual execution environment for software deployment
Bowen Alpern, Joshua Auerbach, Vasanth Bala, Thomas Frauenhofer, Todd Mummert, Michael Pigott
June 2005 **Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments**
Publisher: ACM Press
Full text available: pdf(299.26 KB) Additional Information: full citation, abstract, references, index terms

The Progressive Deployment System (PDS) is a virtual execution environment and infrastructure designed specifically for deploying software, or "assets", on demand while enabling management from a central location. PDS intercepts a select subset of system calls on the target machine to provide a partial virtualization at the operating system level. This enables an asset's install-time environment to be reproduced virtually while otherwise not isolating the asset from peer applications on the targ ...

Keywords: deployment, installation, management, streaming, virtualization

10 GLARE: A Grid Activity Registration, Deployment and Provisioning Framework
Mumtaz Siddiqui, Alex Villazon, Jurgen Hofer, Thomas Fahringer
November 2005 **Proceedings of the 2005 ACM/IEEE conference on Supercomputing SC**

'05

Publisher: IEEE Computer Society

Full text available: pdf(1.43 MB) Additional Information: full citation, abstract, index terms

Resource management is a key concern for implementing effective Grid middleware and shielding application developers from low level details. Existing resource managers concentrate mostly on physical resources. However, some advanced Grid programming environments allow application developers to specify Grid application components at high level of abstraction which then requires an effective mapping between high level application description (activity types) and actual deployed software components ...

11 Article abstracts with full text online: Component evolution and versioning state of the art

Alexander Stuckenholz

January 2005 **ACM SIGSOFT Software Engineering Notes**, Volume 30 Issue 1

Publisher: ACM Press

Full text available: pdf(213.99 KB) Additional Information: full citation, abstract, references, index terms

Emerging component-based software development architectures promise better re-use of software components, greater flexibility, scalability and higher quality of services. But like any other piece of software too, software components are hardly perfect, when being created. Problems and bugs have to be fixed and new features need to be added.This paper analyzes the problem of component evolution and the incompatibilities which result during component upgrades. We present the state of the art in co ...

12 A quantitative analysis of the performance impact of specialized bytecodes in java

Ben Stephenson, Wade Holst

October 2004 **Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research**

Publisher: IBM Press

Full text available: pdf(365.60 KB) Additional Information: full citation, abstract, references, index terms

Java is implemented by 201 bytecodes that serve the same purpose as assembler instructions while providing object-file platform independence. A collection of core bytecodes provide critical and independent functionality while a collection of specialized bytecodes is meant to improve on the performance of some of the core bytecodes. This study identifies 67 specialized bytecodes and shows the impact of their removal by despecializing them into semantically equivalent core bytecodes.

A d ...

13 Using generative design patterns to generate parallel code for a distributed memory environment

Kai Tan, Duane Szafron, Jonathan Schaeffer, John Anvik, Steve MacDonald

June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '03**, Volume 38 Issue 10

Publisher: ACM Press

Full text available: pdf(385.41 KB) Additional Information: full citation, abstract, references, index terms

A design pattern is a mechanism for encapsulating the knowledge of experienced designers into a re-usable artifact. Parallel design patterns reflect commonly occurring parallel communication and synchronization structures. Our tools, CO2P3S (Correct Object-Oriented Pattern-based Parallel Programming System) and MetaCO2P3S, use *generative design patterns*. A programmer selects the parallel design patterns that are appropriate for an application, and then adapts the patterns for that specifi ...

Keywords: design patterns, frameworks, parallel programming, programming tools

Results 1 - 13 of 13

PORTAL

USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+(classpath or "class path" or +directory or +folder) +classe

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **classpath** or **class path** or **directory** or **folder** **classes** **search** **cache**

Found **1** of **184,245**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 1 of 1

Relevance scale

1 Article abstracts with full text online: Component evolution and versioning state of the art

Alexander Stuckenholz

January 2005 **ACM SIGSOFT Software Engineering Notes**, Volume 30 Issue 1

Publisher: ACM Press

Full text available: pdf(213.99 KB) Additional Information: full citation, abstract, references, index terms

Emerging component-based software development architectures promise better re-use of software components, greater flexibility, scalability and higher quality of services. But like any other piece of software too, software components are hardly perfect, when being created. Problems and bugs have to be fixed and new features need to be added.This paper analyzes the problem of component evolution and the incompatibilities which result during component upgrades. We present the state of the art in co ...

Results 1 - 1 of 1

Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

Results for "((classpath<in>metadata) <and> (classes<in>metadata))<and> (search<..."
Your search matched **0** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

e-mail

» Search Options

View Session History

New Search

Modify Search

((classpath<in>metadata) <and> (classes<in>metadata))<and> (search<in>meta

Search »

□ Check to search only within this results set

Display Format: ● Citation ○ Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

No results were found.

Please edit your search criteria and try again. Refer to the Help pages if you need assistance revising your search

IEEE Xplore® RELEASE 2.1

Home | Login | Logout | Access Information | Al

Welcome United States Patent and Trademark Office

Search Results

BROWSE SEARCH IEEE XPLORE GUIDE

Results for **"((classpath)<in>metadata)"**
Your search matched **0** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

e-mail

» Search Options

View Session History

New Search

Modify Search

((classpath)<in>metadata)

Search »

□ Check to search only within this results set

Display Format: ● Citation ○ Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

No results were found.

Please edit your search criteria and try again. Refer to the Help pages if you need assistance revising your search

Help Contact Us Privac

Indexed by Inspec®

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

title: +(classpath or class-path or "class path") SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **classpath** or **class path** or **class path**

Found 8 of 184,245

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 8 of 8

Relevance scale

1 Full papers: Runtime aspect weaving through metaprogramming
Jason Baker, Wilson Hsieh
April 2002 **Proceedings of the 1st international conference on Aspect-oriented software development**
Publisher: ACM Press
Full text available: pdf(883.36 KB) Additional Information: full citation, abstract, references, citings, index terms

We describe an extension to the Java language, Handi-Wrap, that supports weaving aspects into code at runtime. Aspects in Handi-Wrap take the form of method wrappers, which allow aspect code to be inserted around method bodies like advice in AspectJ. Handi-Wrap offers several advantages over static aspect languages such as AspectJ. First, aspects can be woven into binary libraries. Second, a wrapper in Handi-Wrap is a first-class Java value, which allows users to exploit Java mechanisms to defin ...

2 Reviewed papers: Towards an integrated learning laboratory environment for first-year computer science students
Denis Riordan
December 2002 **ACM SIGCSE Bulletin**, Volume 34 Issue 4
Publisher: ACM Press
Full text available: pdf(106.08 KB) Additional Information: full citation, abstract, references

This paper describes an evolving attempt to provide an integrated learning laboratory environment (ILLE) to enhance the learning goals and strategies for first-year computer science students using Java as a first language. The proposed ILLE models a real computer-learning laboratory in which instructors walk around watching and helping students on request. In essence, the system consists of a Java development environment centered round a live communicator with a facility for an instructor to foc ...

3 OOPSLA onward!: Finding bugs is easy
David Hovemeyer, William Pugh
December 2004 **ACM SIGPLAN Notices**, Volume 39 Issue 12
Publisher: ACM Press
Full text available: pdf(506.92 KB) Additional Information: full citation, abstract, references

Many techniques have been developed over the years to automatically find bugs in software. Often, these techniques rely on formal methods and sophisticated program analysis. While these techniques are valuable, they can be difficult to apply, and they aren't always effective in finding real bugs.*Bug patterns* are code idioms that are often errors. We have implemented automatic detectors for a variety of bug patterns found in Java programs. In this paper, we describe how we have used bug pa ...

4 The Java syntactic extender (JSE)
Jonthan Bachrach, Keith Playford
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications OOPSLA '01**, Volume 36 Issue 11
Publisher: ACM Press
Full text available: pdf(198.11 KB) Additional Information: full citation, abstract, references, citings, index terms

The ability to extend a language with new syntactic forms is a powerful tool. A sufficiently flexible macro system allows programmers to build from a common base towards a language designed specifically for their problem domain. However, macro facilities that are integrated, capable, and at the same time simple enough to be widely used have been limited to the Lisp family of languages to date. In this paper we introduce a macro facility, called the Java Syntactic Extender (JSE), with the superio ...

5 Extracting library-based object-oriented applications
Peter F. Sweeney, Frank Tip
November 2000 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th ACM SIGSOFT international symposium on Foundations of software engineering: twenty-first century applications SIGSOFT '00/FSE-8**, Volume 25 Issue 6
Publisher: ACM Press
Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index terms

In an increasingly popular model of software distribution, software is developed in one computing environment and deployed in other environments by transfer over the internet. Extraction tools perform a static whole-program analysis to determine unused functionality in applications in order to reduce the time required to download applications. We have identified a number of scenarios where extraction tools require information beyond what can be inferred through static analysis: software distr ...

6 Practical extraction techniques for Java
Frank Tip, Peter F. Sweeney, Chris Laffra, Aldo Eisma, David Streeter
November 2002 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 24 Issue 6
Publisher: ACM Press
Full text available: pdf(1.01 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Reducing application size is important for software that is distributed via the internet, in order to keep download times manageable, and in the domain of embedded systems, where applications are often stored in (Read-Only or Flash) memory. This paper explores extraction techniques such as the removal of unreachable methods and redundant fields, inlining of method calls, and transformation of the class hierarchy for reducing application size. We implemented a number of extraction techniques in < ...

Keywords: Application extraction, call graph construction, class hierarchy transformation, packaging, whole-program analysis

7
An Eclipse-based environment for molecular simulation

Henrique F. Bucher, Andrew J. Schultz, David A. Kofke
October 2005 **Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange eclipse '05**
Publisher: ACM Press
Full text available: pdf(425.40 KB) Additional Information: full citation, abstract, references, index terms

Etomica is an extensible framework for conducting molecular simulations, and it comprises an API and a graphical IDE based on the *Eclipse* framework. It is written almost entirely in Java, and presently comprises about 1200 classes. In this paper we describe the general structure of the *Etomica* API, and discuss how we have integrated it into the *Eclipse* framework. Topics discussed include the design of the simulation framework and the handling of OpenGL graphics that p ...

Keywords: API, IDE, molecular simulation

8 Service security: A concrete solution for web services adaptability using policies and aspects □
Fabien Baligand, Valérie Monfort
November 2004 **Proceedings of the 2nd international conference on Service oriented computing**
Publisher: ACM Press
Full text available: pdf(368.60 KB) Additional Information: full citation, abstract, references, index terms

Traditional middleware is usually developed on monolithic and non-evolving entities, resulting in a lack of flexibility and interoperability. Among current architectures, Service Oriented Architectures aim to easily develop more adaptable Information Systems. Most often, Web Service is the fitted technical solution which provides the required loose coupling to achieve such architectures. However there is still much to be done in order to obtain a genuinely flawless Web Service, and current ma ...

Keywords: adaptability, aspect oriented programming, reusability, service, service oriented architecture, web service

Results 1 - 8 of 8

org.netbeans.api.java.classpath

Class ClassPath

```
java.lang.Object
  └org.netbeans.api.java.classpath.ClassPath
```

```
public final class ClassPath
extends Object
```

ClassPath objects should be used to access contents of the ClassPath, searching for resources, objects reachable using the ClassPath at runtime. It is intended to replace some of the functionality of org.openide.filesystems.Repository.
ClassPath instances should be used to map from java-style resource names to FileObject (NetBeans-style resource) and vice versa. It should be also used whenever the operation requires inspection of development or runtime project environment instead. The service supports either searching in the classpath resource space, properly hiding resources as the ClassLoader would do at runtime. It can effectively say whether a FileObject is within the reach of a ClassPath or whether it is *reachable* (visible to a ClassLoader). One can translate filenames to resource names and vice versa.

A client may obtain a ClassPath instance using `ClassPath.getClassPath(id)` static method, where the ID is an abstract name for the classpath wanted. There are some predefined classpath names predefined as symbolic constants, following individual types of services (compiler, debugger, executor). Names are not limited to the listed ones; an extension module might add its own private classpath type.

Nested Class Summary

class	`ClassPath.Entry` Represents an individual entry in the ClassPath.

Field Summary

static String	`BOOT` Boot ClassPath of the JDK.
static String	`COMPILE` ClassPath for compiling things.
static String	`DEBUG` **Deprecated.** *Probably useless.*
static String	`EXECUTE` Classpath setting for executing things.

static String	PROP_ENTRIES Name of the "entries" property
static String	PROP_ROOTS Name of the "roots" property
static String	SOURCE ClassPath for project sources.

Method Summary

void	addPropertyChangeListener(PropertyChangeListener l) Adds a property change listener to the bean.
boolean	contains(FileObject f) Convenience method, which checks whether a FileObject lies on this classpath.
List<ClassPath.Entry>	entries() Returns list of classpath entries from the ClassPath definition.
List<FileObject>	findAllResources(String resourceName) Gives out an ordered collection containing all FileObjects, which correspond to a given ResourceName; only the first one is seen by the ClassLoader at runtime or can be linked against.
FileObject	findOwnerRoot(FileObject resource) Finds a root in this ClassPath, that owns the given file.
FileObject	findResource(String resourceName) Returns a FileObject for the specified resource.
ClassLoader	getClassLoader(boolean cache) Returns a ClassLoader for loading classes from this ClassPath.
static ClassPath	getClassPath(FileObject f, String id) Find the classpath of a given type, if any, defined for a given file.
String	getResourceName(FileObject f) Creates a suitable resource name for the given FileObject within the classpath.
String	getResourceName(FileObject f, char dirSep, boolean includeExt) Computes a resource name for the FileObject, which uses `pathSep' character as a directory separator.
FileObject[]	getRoots() Retrieves valid roots of ClassPath, in the proper order.
boolean	isResourceVisible(FileObject resource) Determines if the resource is *visible* in the classpath, that is if the file will be reached when a process attempts to load a resource of that name.
void	removePropertyChangeListener(PropertyChangeListener l) Removes the listener registered by `addPropertyChangeListener/`
String	toString()

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

EXECUTE

```
public static final String EXECUTE
```

Classpath setting for executing things. This type can be used to learn runtime time classpath for execution of the file in question.

It corresponds to the `-classpath` option to `java` (the Java launcher): i.e. all compiled classes outside the JRE that will be needed to run the program, or at least to load a certain class. It may also be thought of as corresponding to the list of URLs in a `URLClassLoader` (plus URLs present in parent class loaders but excluding the bootstrap and extension class loaders).

See Also:
Constant Field Values

DEBUG

```
public static final String DEBUG
```

Deprecated. *Probably useless.*
Classpath for debugging things

See Also:
Constant Field Values

COMPILE

```
public static final String COMPILE
```

ClassPath for compiling things. This type can be used to learn compilation time classpath for the file in question.

It corresponds to the `-classpath` option to `javac`: i.e. already-compiled classes which some new sources need to compile against, besides what is already in the JRE.

See Also:
Constant Field Values

SOURCE

```
public static final String SOURCE
```

ClassPath for project sources. This type can be used to learn package root of the file in question.

It is similar to the `-sourcepath` option of `javac`.

For typical source files, the sourcepath will consist of one element: the package root of the source file. If more than one package root is to be compiled together, all the sources should share a sourcepath with multiple roots.

Note that each source file for which editor code completion (and similar actions) should work should have a classpath of this type.

Since:
org.netbeans.api.java/1 1.4
See Also:
Constant Field Values

BOOT

```
public static final String BOOT
```

Boot ClassPath of the JDK. This type can be used to learn boot classpath which should be used for the file in question.

It corresponds to the `-Xbootclasspath` and `-Xext` options to `java` (the Java launcher): i.e. all compiled classes in the JRE that will be needed to run the program. It may also be thought of as corresponding to the classes loadable by the primordial bootstrap class loader *plus* the standard extension and endorsed-library class loaders; i.e. class loaders lying below the regular application startup loader and any custom loaders. Generally there ought to be a single boot classpath for the entire application.

Since:
org.netbeans.api.java/1 1.4
See Also:
Constant Field Values

PROP_ROOTS

```
public static final String PROP_ROOTS
```

Name of the "roots" property

See Also:
Constant Field Values

PROP_ENTRIES

```
public static final String PROP_ENTRIES
```

Name of the "entries" property

See Also:
Constant Field Values

Method Detail

getRoots

```
public FileObject[] getRoots()
```

Retrieves valid roots of ClassPath, in the proper order. If there's an entry in the ClassPath, which cannot be accessed, its root is not returned by this method. FileObjects returned are all folders.

Returns:
array of roots (folders) of the classpath. Never returns null.

---

entries

```
public List<ClassPath.Entry> entries()
```

Returns list of classpath entries from the ClassPath definition. The implementation must ensure that modifications done to the List are banned or at least not reflected in other Lists returned by this ClassPath instance. Clients must assume that the returned value is immutable.

Returns:
list of definition entries (Entry instances)

---

findResource

```
public final FileObject findResource(String resourceName)
```

Returns a FileObject for the specified resource. May return null, if the resource does not exist, or is not reachable through this ClassPath.
If the *resourceName* identifies a package, this method will return the `FileObject` for the first *package fragment* in the `ClassPath` Note: do not pass names starting with slash to this method.

Parameters:

`resourceName` - name of the resource as it would be passed to `ClassLoader.getResource(java.lang.String)`

Returns:

FileObject for the resource, or null if the resource cannot be found in this ClassPath.

---

findAllResources

```
public final List<FileObject> findAllResources(String resourceName)
```

Gives out an ordered collection containing all FileObjects, which correspond to a given ResourceName; only the first one is seen by the ClassLoader at runtime or can be linked against. The resource name uses slashes ('/') as folder separator and must not start with slash.

Parameters:

`resourceName` - resource name

Returns:

list of resources identified by the given name.

---

getResourceName

```
public final String getResourceName(FileObject f)
```

Creates a suitable resource name for the given FileObject within the classpath. The method will return `null` if the fileobject is not underneath any of classpath roots.
The returned name uses uses slashes ('/') as folder separators and dot ('.') to separate file name and its extension. Note that if the file object is in the classpath subtree, but is not reachable (it is hidden by some other resource), the resource name is still returned.

Parameters:

`f` - FileObject whose resource name is requested

Returns:

Java-style resource name for the given file object (the empty string for the package root itself), or null if not within the classpath

---

getResourceName

```
public final String getResourceName(FileObject f,
                                    char dirSep,
                                    boolean includeExt)
```

Computes a resource name for the FileObject, which uses `pathSep' character as a directory separator. The resource name can be returned without the file extension, if desired. Note that parent folder names are always returned with extension, if they have some.

Parameters:

`f` - FileObject whose resource name is requested.

`dirSep` - directory separator character

`includeExt` - whether the FileObject's extension should be included in the result

Returns:

resource name for the given FileObject (the empty string for the package root itself) or null

---

findOwnerRoot

```
public final FileObject findOwnerRoot(FileObject resource)
```

Finds a root in this ClassPath, that owns the given file. File resources, that are not reachable (they are hidden by other resources) are still considered to be part of the classpath and "owned" by one of its roots.
Note: This implementation assumes that the FileSystem hosting a classpath root contains the entire classpath subtree rooted at that root folder.

Parameters:

`resource` - resource to find root for.

Returns:

classpath root, which hosts the specified resouce. It can return null, if the resource is not within the ClassPath contents.

---

contains

```
public final boolean contains(FileObject f)
```

Convenience method, which checks whether a FileObject lies on this classpath. It is an equivalent of `getResourceName(f) != null`

Parameters:

`f` - the FileObject to check

Returns:

true, if the parameter is inside one of the classpath subtrees, false otherwise.

---

isResourceVisible

```
public final boolean isResourceVisible(FileObject resource)
```

Determines if the resource is *visible* in the classpath, that is if the file will be reached when a process attempts to load a resource of that name. It will return false when the resource is not contained in the classpath.

Parameters:

`resource` - the resource whose visibility should be tested

Returns:

true, if the resource is contained in the classpath and visible; false otherwise.

---

addPropertyChangeListener

```
public final void addPropertyChangeListener(PropertyChangeListener l)
```

Adds a property change listener to the bean.

---

removePropertyChangeListener

```
public final void removePropertyChangeListener(PropertyChangeListener l)
```

Removes the listener registered by `addPropertyChangeListener`/

---

getClassPath

```
public static ClassPath getClassPath(FileObject f,
                                     String id)
```

Find the classpath of a given type, if any, defined for a given file.

This method may return null, if:

- the path type (`id` parameter) is not recognized
- the path type is not defined for the given file object

Generally you may pass either an individual Java file, or the root of a Java package tree, interchangeably, since in most cases all files in a given tree will share a single classpath.

Typically classpaths for files are defined by the owning project, but there may be other ways classpaths are defined. See `ClassPathProvider` for more details.

Parameters:

`f` - the file, whose classpath settings should be returned (may *not* be null as of org.netbeans.api.java/1 1.4)

`id` - the type of the classpath (e.g. `COMPILE`)

Returns:

classpath of the desired type for the given file object, or `null`, if there is no classpath available

See Also:

`ClassPathProvider`

toString

```
public String toString()
```

Overrides:

`toString` in class `Object`

getClassLoader

```
public final ClassLoader getClassLoader(boolean cache)
```

Returns a ClassLoader for loading classes from this ClassPath.

If `cache` is false, then the method will always return a new class loader. If that parameter is true, the method may return a loader which survived from a previous call to the same `ClassPath`.

Parameters:

`cache` - true if it is permissible to cache class loaders between calls

Returns:

class loader which uses the roots in this class path to search for classes and resources

Since:

1.2.1

Overview Package Class Use Tree Deprecated Index Help *org.netbeans.api.java/1 1.11*
PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Sign in

Web Images Video^New!^ News Maps **more »**

+(classpath or "class path" or +directory or +f(Search Advanced Search
Preferences

Lowercase "or" was ignored. Try "OR" to search for either of two terms. [details]

Web Results **11** - **20** of about **31,300** for **+(classpath or "class path" or +directory or +folder) +classes +se**

Deployment automation
Run the application several times, and check the JIT **cache directory** size ... Note: The java.**class.path** system property is set by JetPackII automatically. ...
www.excelsior-usa.com/doc/jet450/jetw006.html - 126k - Cached - Similar pages
[More results from www.excelsior-usa.com]

Portal Content Management
Use true to have the DocumentManager **cache search** results in the ... .jar into a .jar file referenced in the META-INF/Manifest.mf file's **Class-Path** entry in ...
edocs.bea.com/wlp/docs70/dev/conmgmt.htm - 152k - Cached - Similar pages
[More results from edocs.bea.com]

TWiki . Netbeans . FaqJdk11
ZIP; C:\WINDOWS\Java\classes;C:\WINDOWS\Java\trustlib'/> <property name='java.specification.version' value='1.1'/> <property name='java.**class.path**' ...
wiki.java.net/bin/view/Netbeans/FaqJdk11 - 26k - Cached - Similar pages

KineticFusion Properties Details
The system **class path** is a good location to place the V2 component **classes**, if they are used by the user. The installation **classes directory** contains a set ...
www.kinesissoftware.com/products/docs/10-config/propdetails.html - 120k - Cached - Similar pages

KineticFusion Properties Summary
storeDuplicateSystemClasses: Should KineticFusion decompile AS2.0 **classes** that already exist on the system **class path**? kinesis.actionscript. ...
www.kinesissoftware.com/products/docs/10-config/summary.html - 22k - Cached - Similar pages

Frequently Asked Questions
Next you will need some xml and configuration files in your **class path** for OJB. ...
Therefore, it is easiest if you put them in the WEB-INF/**classes folder** ...
db.apache.org/ojb/docu/faq.html - 86k - Cached - Similar pages

Java Custom Tag
should be placed in a CF Java **class path**. You can set multiple classpaths in the ColdFusion Administrator, so just map a **classpath** to the **folder** where the ...
www.houseoffusion.com/groups/CF-Talk/thread.cfm/threadid:10911 - 31k - Cached - Similar pages

Eclipse runtime options
This property may also be set to a comma-separated **class path** entries which ... then a **search** is done in parent **directory** of the org.eclipse.osgi bundle. ...
help.eclipse.org/help31/topic/org.eclipse.platform.doc.isv/reference/misc/runtime-options.html - 38k - Cached - Similar pages

Eclipse runtime options
This property may also be set to a comma-separated **class path** entries which are added to ... The value "**folder**" indicates that the Eclipse OSGi Framework is ...

help.eclipse.org/help32/topic/org.eclipse.platform.doc.isv/reference/misc/runtime-options.html - 44k - Cached - Similar pages

Customizing IBM Rational ProjectConsole to modify default font ...
The author created a **folder** titled "PjC" at the root level, which is where the three JAR files are stored. ... Publisher **Class-Path**: ..\\Classes\\klg45.jar ...
www-128.ibm.com/developerworks/rational/library/05/503_pjc/ - 56k -
Cached - Similar pages

◀ Gooooooooooogle ▶
Result Page: **Previous** 1 **2** 3 4 5 6 7 8 9 10 11 **Next**

+(classpath or "class path" or +direc Search

Search within results | Language Tools | Search Tips

Google Home - Advertising Programs - Business Solutions - About Google

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	4	("5187786" or "20040133882"). pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:13
L2	6	("5187786" or "20040133882" "6212564").pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:14
L3	14	java and (classpath or class-path or (class adj path)) and classes! and (search with cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:30
L4	398	java and (classpath or class-path or (class adj path)) and classes!	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:35
L5	100	4 and cache and search$6 and (locate$1 or locator or locating)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:37
L6	58	4 and cache and search$6 and (locate$1 or locator or locating) and ((directory or folder) with class)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:44
L7	16	6 and (search$6 or locate or locating or find$4 or retriev or extract$5) with (class and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:45
L8	16	6 and (search$6 or locate$1 or locating or find$4 or retriev or extract$5) with (class and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:45
L9	34	6 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5) with (class and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:45

L10	39	6 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) with (class and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:54
L11	39	10 and (wrapper or interfac$3 or API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:50
L12	57	4 and wrapper and interfac$3 and (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:51
L13	10	12 and wrapper same (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:19
L14	33	12 and wrapper same (interfac$3 or API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:53
L15	20	12 and wrapper with (interfac$3 or API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:54
L16	13	15 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) and (class and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:56
L17	0	4 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) adj9 ((class adj libary) and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 06:57
L18	0	4 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) adj9 ((class adj library) and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:15
L19	4	4 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) adj9 ((class adj library))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:19

L20	129	java and (classpath or class-path or (class adj path)) and classes! and (search$5 and cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:18
L21	6	20 and wrapper same (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:27
L22	97	20 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) adj9 (class or (class adj library))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:25
L23	41	22 and (search$6 or locate$1 or locating or find$4 or retriev$4 or extract$5 or quer$5 or request$3) adj9 (cach$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:20
L24	62	22 and (classloader or (class adj loader))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:25
L25	10	4 and wrapper same (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:28
L26	29	4 and (wrapper same interface) and (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:30
L27	3	26 and zip adj file	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:30
L28	1078	(wrapper same interface) and (API or (application adj programming adj interface))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:30
L29	18	28 and zip adj file	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:30

EAST Search History

L30	6	29 and java and ((classpath or class-path or (class adj path) or directory or folder) with class) and search$6 and request$3 and cach$4	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:32
L31	11	29 and java and ((classpath or class-path or (class adj path) or directory or folder) and class) and search$6 and request$3 and cach$4	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:33
L32	170	28 and java and ((classpath or class-path or (class adj path) or directory or folder) and class) and search$6 and request$3 and cach$4	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:33
L33	37	28 and java and ((classpath or class-path or (class adj path) or directory or folder) with class) and search$6 and request$3 and cach$4	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:34
L34	15	33 and java with wrapper	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:35
L35	18	33 and java same wrapper	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:39
L36	37	33 and wrapper	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/08/25 07:39